

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application No.: 10/092,168  
Filed: March 6, 2002  
Inventor:  
Sridhar Satuloori, et al.

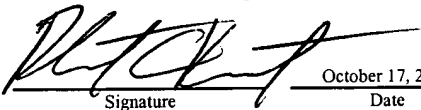
Examiner: Zhen, Li B.  
Group/Art Unit: 2194  
Atty. Dkt. No: 5681-08800

Title: APPLICATION PROGRAMS  
WITH DYNAMIC  
COMPONENTS

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on the date indicated below.

Robert C. Kowert

Name of Registered Representative

  
Signature

October 17, 2005

Date

**PRE-APPEAL BRIEF REQUEST FOR REVIEW**

**Mail Stop AF**

Commissioner for Patents  
P.O. Box 1450  
Alexandria, VA 22313-1450

Dear Sir:

Applicant requests review of the final rejection in the above-identified application. No amendments are being filed with this request. This request is being filed with a notice of appeal. The review is requested for the reason(s) stated below.

Claims 1-53 remain pending in the application. Reconsideration of the present case is earnestly requested in light of the following remarks. Please note that for brevity, only the primary arguments directed to the independent claims are presented, and that additional arguments, e.g., directed to the subject matter of the dependent claims, will be presented if and when the case proceeds to Appeal. Claims 1-53 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Carlson et al. (U.S. Publication 2003/0056022) (hereinafter "Carlson") in view of "3 The Model-View-Controller Architecture" (hereinafter "3MVC"). Applicants note the following clear errors in the Examiner's rejection.

**Applicants submit that the Examiner has failed to provide a *prima facie* rejection of claims 1, 14, 27 and 41.** Regarding claim 1, contrary to the Examiner's assertion, Carlson in view of MVC fails to teach or suggest a dynamic component generator configured to receive a new set of requirements for an application. Carlson teaches configurable JAVA classes created as instances of a metaclass object that includes subclasses and interfaces that themselves include methods to alter attributes and methods of a

configurable JAVA class. MVC teaches the decoupling of model, view and controller objects of an application. The Examiner cites passages of Carlson (col. 5:14-25) describing how Carlson's "invention allows the creation of new Java classes and the change of existing Java classes" and that "new functionality can be introduced by configuring new classes rather than redevelopment." However, Carlson in view of MVC fails to teach or suggest a *dynamic component generator* configured to *receive a new set of requirements for an application*. Carlson teaches a metaclass object that includes methods "to alter the attributes and methods of the Java class instance of the metaclass object" (Carlson, paragraph 0025). However, simply providing a system that includes methods to alter attributes and methods of a Java class instance is not the same as an application having a component generator that *receives a new set of requirements for an application*. Carlson does not mention any component of his system receiving a set of requirements for an application. In fact, Carlson is completely silent about communicating application-level requirements. Carlson's system pertains only to a system in which a programmer may utilize Carlson's metaclass object, and specifically the metaclass object's method for altering specific attributes and methods, to modify particular configurable Java classes at runtime.

In response to the above arguments, the Examiner cites paragraphs [0037] and [0041] of Carlson. Paragraph [0037] of Carlson describes Carlson's Softtype Interface 20 and SofttypeBean 22. Carlson teaches that the Softtype Interface 20 allows run-time configuration of the Softtype instances and that the specific methods and attributes of the SofttypeBean 22 can be dynamically defined and altered. Specific properties and methods may be added to instances of SofttypeBean 22. Thus, paragraph [0037] describes the ability to dynamically alter specific properties and methods of existing SofttypeBean instances, but makes no mention of a dynamic component generator configured to *receive a new set of requirements for the application*. Thus, Carlson teaches that properties and methods of class and object instances can be altered but does not teach or suggest anything about receiving a set of *application requirements*.

The Examiner also cites paragraph [0041] where Carlson describes that method and property definitions may be included in an XML file. As illustrated by the Examiner's cited passages, Carlson's system includes dynamically altering specific properties and methods of existing object instantiations according to XML-based *definitions* of properties and methods. Method and property definitions are not the same as application-level requirements. In the Advisory Action, the Examiner again argues that Carlson's definitions of properties and methods correspond to a set of application requirements. Applicants' maintain that Carlson's property and method definitions are not the same as, nor do they include, application requirements. Carlson's property and method definitions are specifications for property types and the method definitions that define interfaces to new methods and classes. However,

Carlson does include a specification for any application-level logic in the property and method definitions. In contrast, application requirements include high-level specifications of application functionality. For instance, Applicants' specification (page 12, lines 10-15) illustrates an example of application requirements. This example includes specifications, not only for data columns of a payroll table, such as "hra" and "basic", but also for calculations used on the table, such as line four of the example that specifies, "use  $\text{hra} = 0.02 * \text{basic}$ ", thus providing a high-level specification of a calculation that should be used with an application. Similarly, Applicants' specification includes another example of application requirements on page 13, which includes a specification for application behavior for a "Balance" method. In this example, the application requirements specify that the method "Balance" should perform a "recompute" action and also specifies that the "recompute" action should perform a particular calculation, "CA+SA+FD+RD". Note that application requirements as described in the specification are not specific method and attribute *definitions*, but instead are high-level functional requirements. Application requirements specify high-level application logic regarding particular application behavior. Carlson's system does not include any such application requirements. Instead, Carlson has provided a particular metaclass that allows modifications, such as the addition of new methods, to instances of the metaclass according to an XML schema defining the interfaces to the new methods and properties.

Carlson in view of MVC also fails to teach or suggest a dynamic component generator configured to generate a second dynamic component to replace the first dynamic component, where the second dynamic component is configured to function according to the new set of requirements. The Examiner states that Carlson's system "allows the creation of new Java classes and the change of existing Java classes" and that in Carlson's system "new functionality can be introduced by configuring new classes". However, the Examiner does not show any passage of the cited art that teaches or suggests a component generator configured to generate a second dynamic component configured to function according to the new set of requirements. Carlson does not describe anything in his system that is configured to generate a component configured to function according to a *new set of requirements* and to *replace a first dynamic component*. As noted above, merely providing methods to alter attributes and methods of Java classes does not imply a component configured to generate new components that function according to a set of requirements. Carlson's system includes functions, such as "addmethod" and "getmethod" that allow the methods of an existing class to be changed, but Carlson's system does not generate new replacement components that function according to a new set of application requirements.

In the Response to Arguments of the Final Action, the Examiner cites paragraph [0036] of Carlson and refers to the fact that Carlson's Softtype object includes a DynamicEntity that includes methods to add and remove directly contained properties and also to add and remove methods from a class object. The Examiner has cited portions of Carlson that describe the ability to modify properties and methods of existing classes and objects according to property and method definitions. As discussed above, property and method definitions are not new application requirements. The Examiner's cited passages do not describe generating a second dynamic component to replace a first dynamic component where the second dynamic component is configured to function according to a new set of requirements for the application. As described above, Carlson teaches altering existing properties and methods according to property and method definitions (Carlson, paragraphs [0035] and [0040-0042]), but nowhere does Carlson teach generating a second dynamic component to *replace* a first dynamic component where the second dynamic component is configured to function *according to a new set of requirements for the application*.

Furthermore, MVC, alone or in combination with Carlson, fails to teach or suggest anything about receiving a set of new application requirements or about generating a dynamic component configured to function according to the new set of application requirements and thus fails to overcome any of the above noted deficiencies of Carlson.

Thus, for at least the reasons presented above, the rejection of claim 1 is not supported by the cited prior art and removal thereof is respectfully requested. Similar remarks as those above regarding claim 1 also apply to claims 14, 27 and 41.

The Examiner's rejection of many of the dependent claims is additionally erroneous. For example, in regards to claim 11, Carlson in view of MVC fails to teach or suggest wherein the first application module is a model module, wherein the static component is a static data model configured to function independent of an application data representation, and wherein the dynamic component is a dynamic data model configured to function dependent upon the application data representation and according to a current set of application requirements in response to the user input. The Examiner cites page 1, paragraph 4 of MVC referring to the fact that under MVC's system, "data are accessed and manipulated through methods that are independent of the GUI". However, nowhere does MVC mention a module that comprises a static data model and a dynamic data model. The Examiner is relying upon Carlson to teach a system including both static and dynamic attributes and methods. However, even in the Examiner's proposed combination, Carlson in view of MVC fail to include a static data model

configured to function *independent of an application data representation* and a dynamic data model configured to function *dependent upon the application data representation*. The Examiner has not cited any portion of Carlson or MVC that describes two such data models, one that functions independent of an application data model and another that functions dependent upon the application data model. Instead, the Examiner relies only on the fact that MCV mentions that "data are accessed and manipulated through methods that are independent of the GUI." A GUI is not an application data representation. A GUI is an interface for receiving user input. Thus MVC simply teaches that data are manipulated by methods that are not a part of a GUI. Thus, the Examiner's combination of Carlson in view of MVC clearly fails to teach or suggest a static data model configured to function independent of an application data representation and a dynamic data model configured to function dependent upon the application data representation. As such, the rejection of claim 11 is not supported by the prior art and removal thereof is respectfully requested. Similar arguments also apply to claims 24, 38 and 51.

In light of the foregoing remarks, Applicant submits the application is in condition for allowance, and notice to that effect is respectfully requested. If any extension of time (under 37 C.F.R. § 1.136) is necessary to prevent the above referenced application from becoming abandoned, Applicants hereby petition for such an extension. If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert & Goetzel PC Deposit Account No. 501505/5681-08800/RCK.

Also enclosed herewith are the following items:

- ☒ Return Receipt Postcard
- ☒ Notice of Appeal

Respectfully submitted,



Robert C. Kowert  
Reg. No. 39,255  
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.  
P.O. Box 398  
Austin, TX 78767-0398  
Phone: (512) 853-8850

Date: October 17, 2005